机器学习
第3节

涉及知识点：
个性化推荐简介、k近邻算法、基于近邻算法的协同过滤、矩阵分解的协同过滤算法、因子分解机

k近邻算法与双线性模型

张伟楠 – 上海交通大学

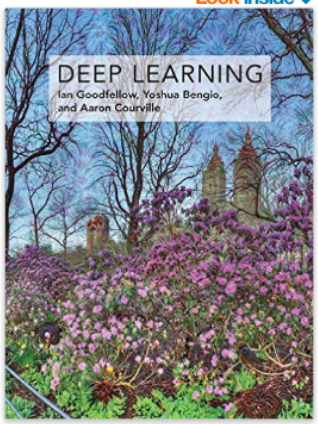
个性化推荐简介

张伟楠 – 上海交通大学

推荐系统

- 个性化推荐是一种广义的信息检索
  - 没有用户查询
  - 用户的信息需求是隐含的

书籍推荐



See this image

Deep Learning (Adaptive Computation and Machine Learning series) Hardcover – November 18, 2016

by Ian Goodfellow (Author), Yoshua Bengio (Author), Aaron Courville (Author)

4.4 out of 5 stars 46 customer reviews

#1 Best Seller in Artificial Intelligence & Semantics

See all formats and editions

Hardcover
$64.00

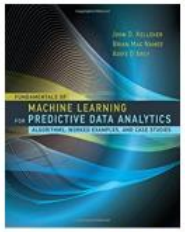
11 Used from $80.12
15 New from $64.00

"Written by three experts in the field, *Deep Learning* is the only comprehensive book on the subject." -- **Elon Musk**, cochair of OpenAI; cofounder and CEO of Tesla and SpaceX

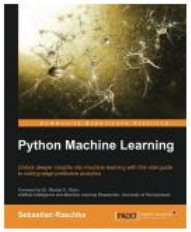
Deep learning is a form of machine learning that enables computers to learn from experience and understand the world in terms of a hierarchy of concepts. Because the computer gathers knowledge from experience, there is no need for a human computer operator to formally specify all the knowledge

Read more

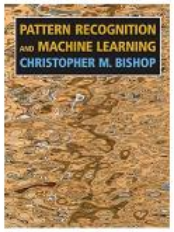
Customers Who Bought This Item Also Bought



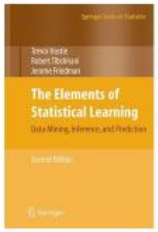
Fundamentals of Machine Learning for Predictive Data Analytics:...
› John D. Kelleher
4.5 out of 5 stars 22
Hardcover
$76.00 Prime



Python Machine Learning
› Sebastian Raschka
4.4 out of 5 stars 98
#1 Best Seller in Computer Neural Networks
Paperback
$40.49 Prime



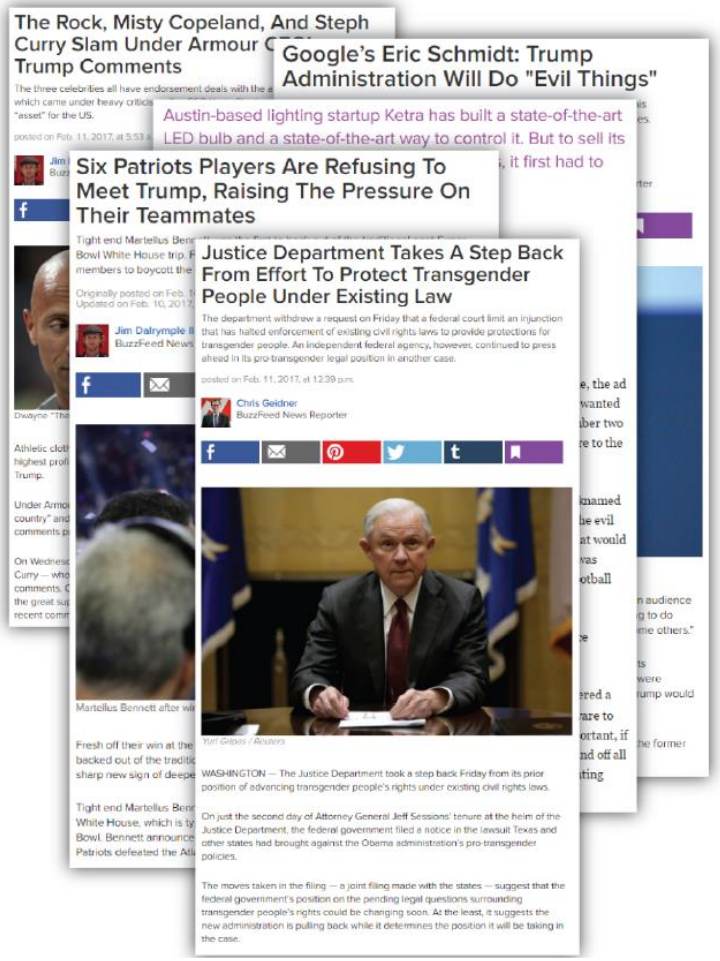
Pattern Recognition and Machine Learning (Information Science and...
› Christopher M. Bishop
4.4 out of 5 stars 132
Hardcover
$63.68 Prime



The Elements of Statistical Learning: Data Mining, Inference, and...
Trevor Hastie
4.4 out of 5 stars 92
#1 Best Seller in Bioinformatics
Hardcover
$73.18 Prime

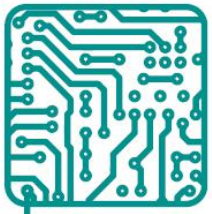
Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques...
› Aurélien Géron
Paperback
$28.56 Prime

新闻流推荐

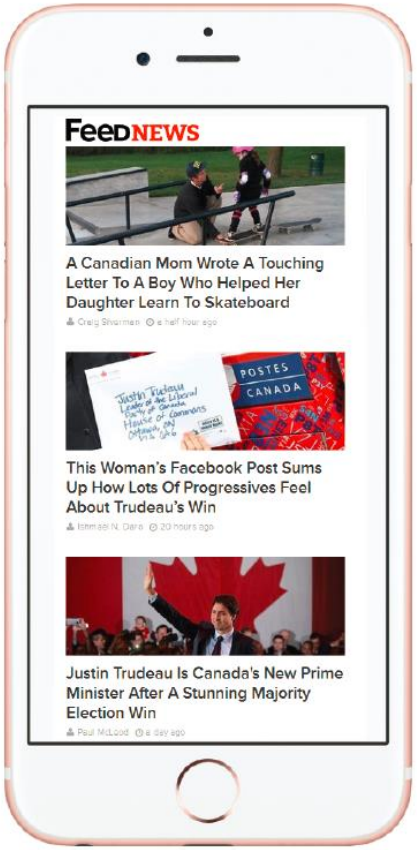


每天有大量的候选文章

推荐系统



编辑人工筛选优质文章



选择优质文章向终端用户提供新闻流

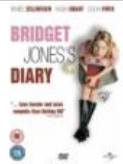
个性化方法

- 根据用户以前喜欢的电影，如何推荐更多用户想要的电影?
  - 方法1：推荐与用户喜欢的电影有共同的演员/导演/流派的电影
  - 方法2：推荐与用户有类似兴趣的用户喜欢的电影

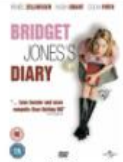
信息过滤

- 信息过滤处理用户可能感兴趣或有用的信息的传递
  - 推荐系统：以建议的形式进行信息过滤
  - 两种信息过滤方法
    - 基于内容的过滤
      - 推荐与用户喜欢的电影共享演员/导演/流派的电影
    - 协同过滤（Collaborative filtering）
      - 推荐与用户有类似兴趣的用户喜欢的电影

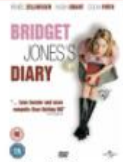
一个（小的）评分矩阵

	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★☆	★★★★★			★☆☆☆☆		
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

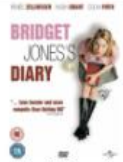
用户

	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★☆	★★★★★			★☆☆☆☆		
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

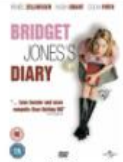
物品

	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★☆	★★★★★			★☆☆☆☆		
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

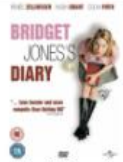
用户-物品评分

	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★☆	★★★★★			★☆☆☆☆		
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

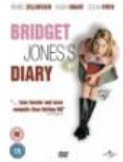
用户画像

	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★☆	★★★★★			★☆☆☆☆		
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

物品画像

	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★☆	★★★★★			★☆☆☆☆		
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

一个空的评分条目

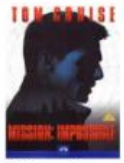
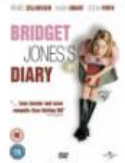
	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★★	★★★★★			★☆☆☆☆		
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

- 在显式评分数据上的推荐
  - 预测空评分

基于近邻算法的协同过滤

张伟楠 – 上海交通大学

一个空的评分条目

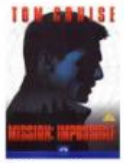
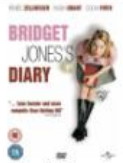
	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★★	★★★★★			★☆☆☆☆		
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

- 在显式数据上的推荐
  - 预测空评分

如果我看过Love Actually，我应该怎么给它打分？

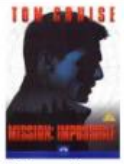
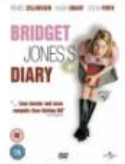
(slide credit: Jun Wang)

一个空的评分条目

	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★☆	★★★★★			★☆☆☆☆		?
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

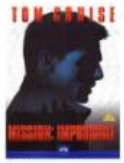
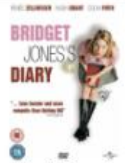
- 你觉得它的评分会是多少?

基于用户的KNN解决方法

	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★☆	★★★★★			★☆☆☆☆		?
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

- 寻找Boris的类似用户（邻居）
  - Dave和George

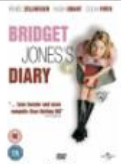
评分预测

	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★☆	★★★★★			★☆☆☆☆		?
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

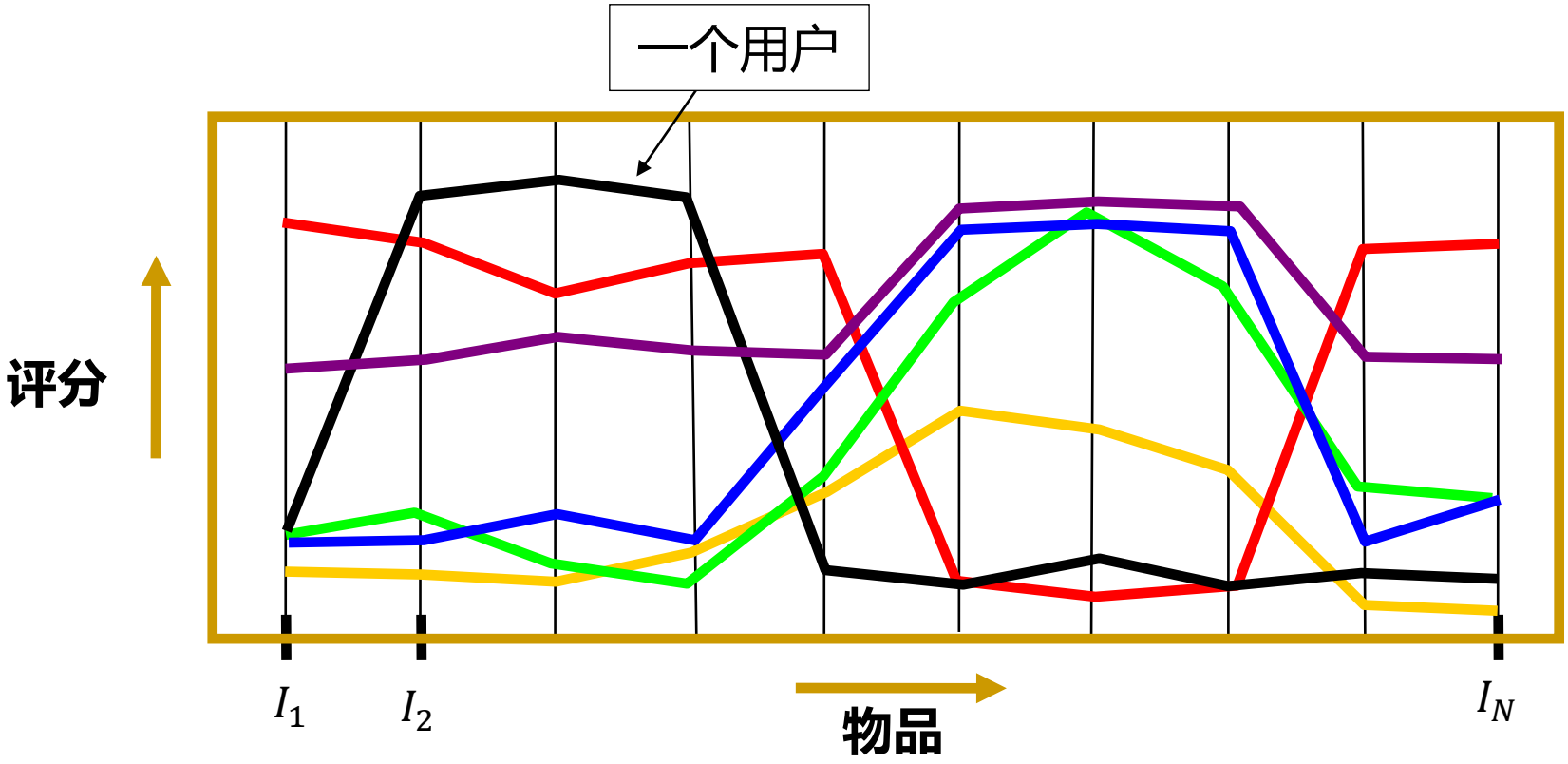
- 平均Dave和George在Love Actually上的评分
  - 预测 = (1+2)/2 = 1.5

用于推荐的协同过滤

- 基于用户的基本KNN算法
  - 为每个目标用户提供推荐
    - 寻找类似用户
    - 基于类似用户推荐新物品

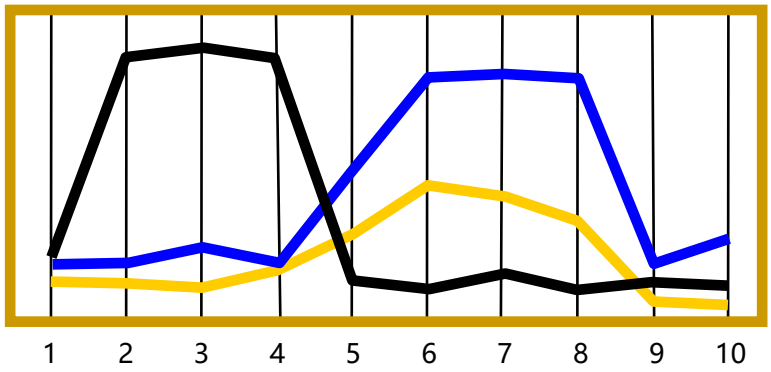
	 Die Hard	 Mission: Impossible	 GoldenEye	 Casino Royale	 Titanic	 Notting Hill	 Bridget Jones's Diary	 Love Actually
Boris	★★★☆☆	★★★★☆	★★★★★			★☆☆☆☆		?
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

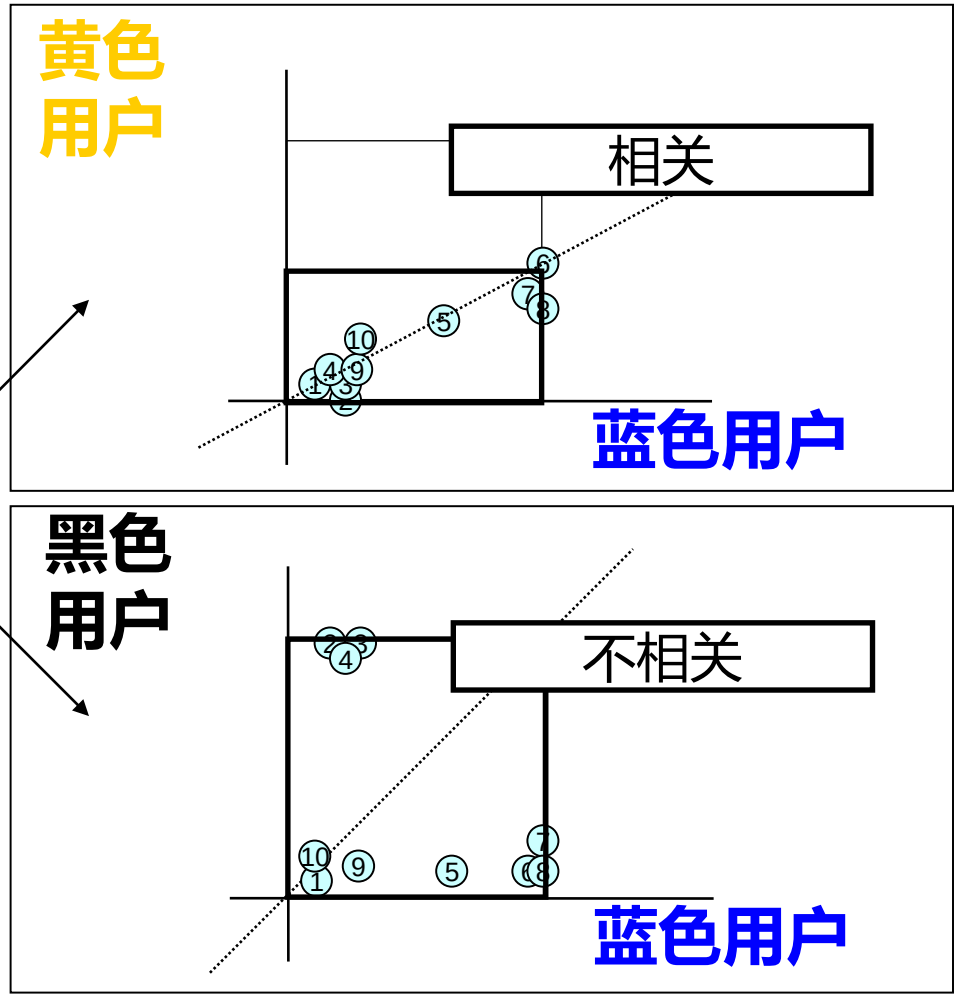
用户的相似度



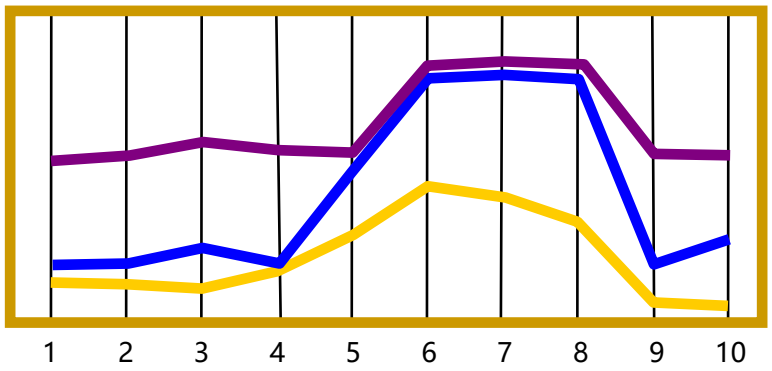
- 每个用户的画像可以根据其选择的物品评级直接构建为向量

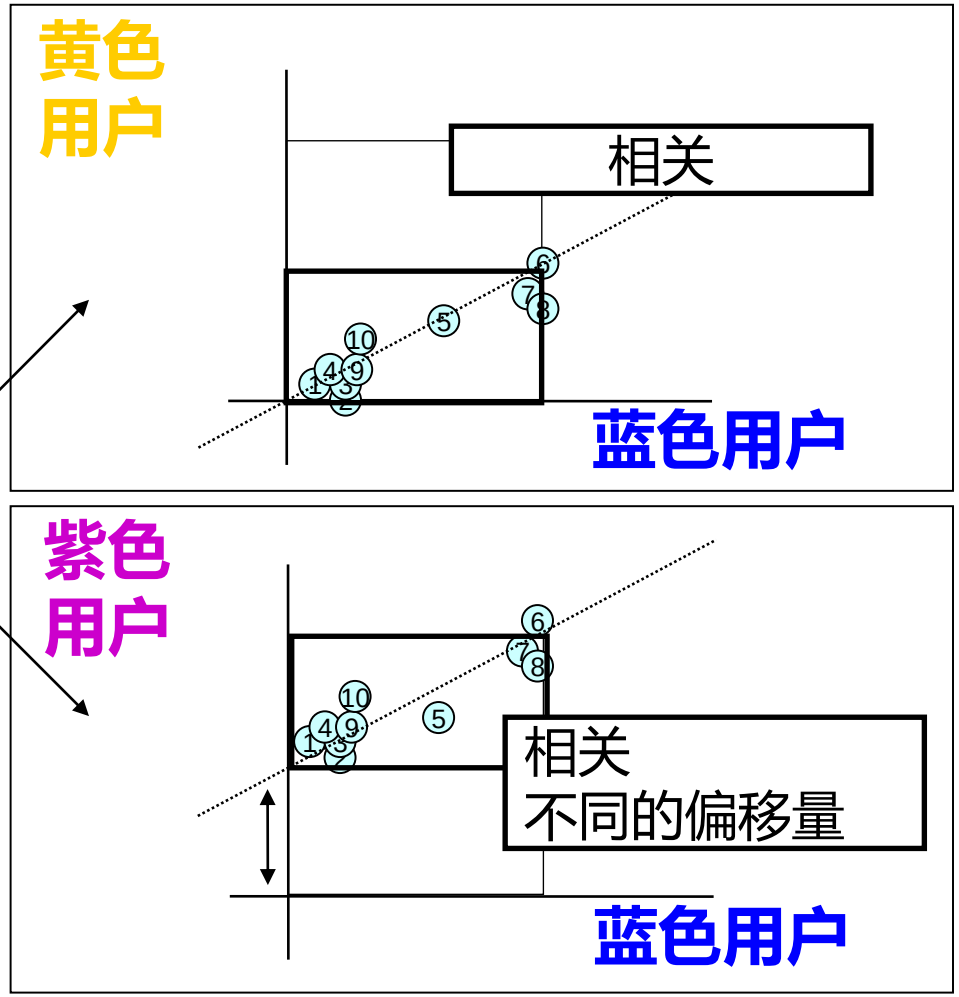
用户的相似度





用户的相似度





相似度量（用户）

- 两个用户a和b之间的相似度量
  - 余弦（角）相似度

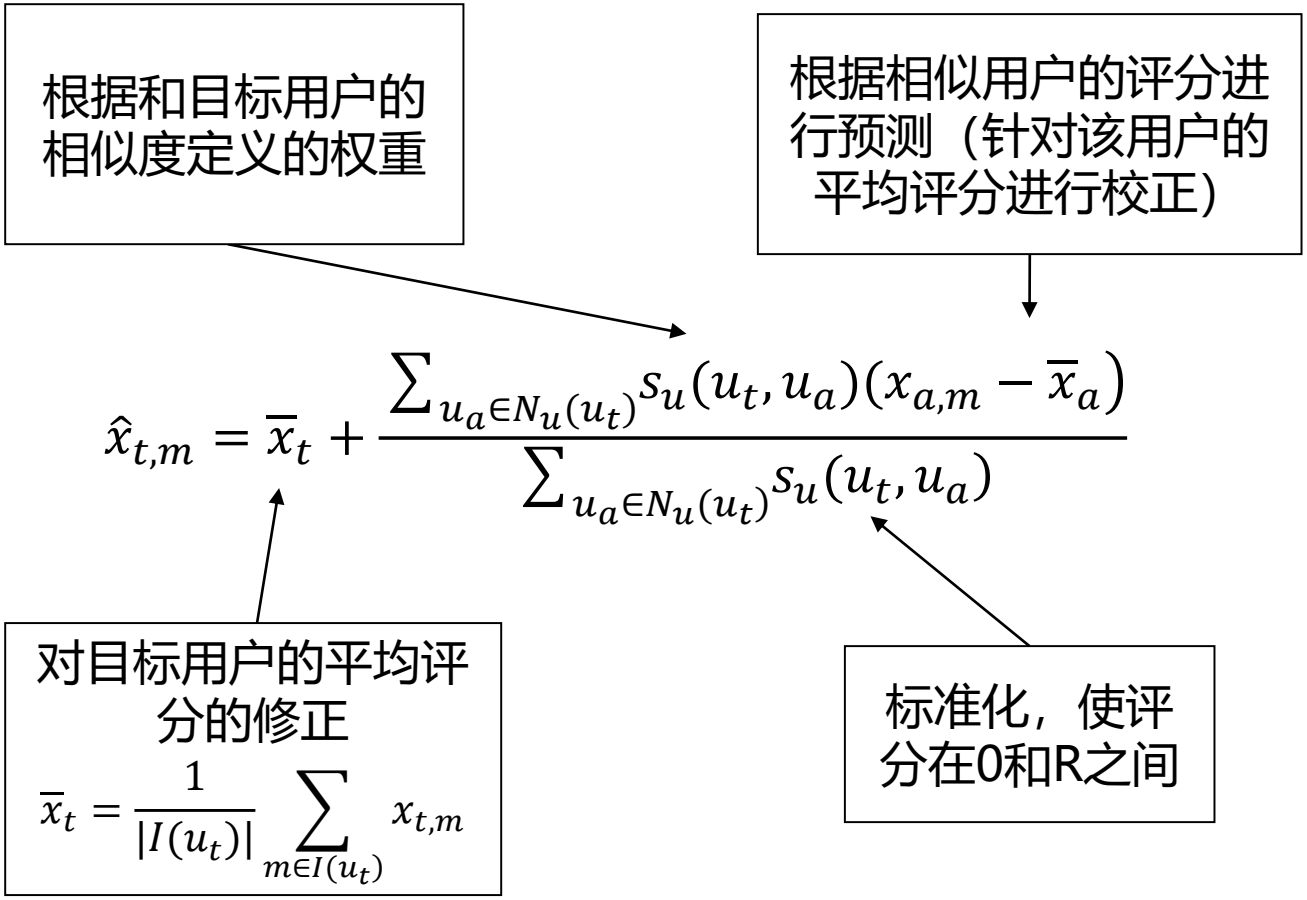
$$s_u^{cos}(u_a, u_b) = \frac{u_a^{\top} u_b}{\| u_a \| \| u_b \|} = \frac{\sum_m x_{a,m} x_{b,m}}{\sqrt{\sum_m x_{a,m}^2 \sum_m x_{b,m}^2}}$$

  - 皮尔森相关性

$$s_u^{corr}(u_a, u_b) = \frac{\sum_m (x_{a,m} - \overline{x}_a)(x_{b,m} - \overline{x}_b)}{\sqrt{\sum_m (x_{a,m} - \overline{x}_a)^2 \sum_m (x_{b,m} - \overline{x}_b)^2}}$$

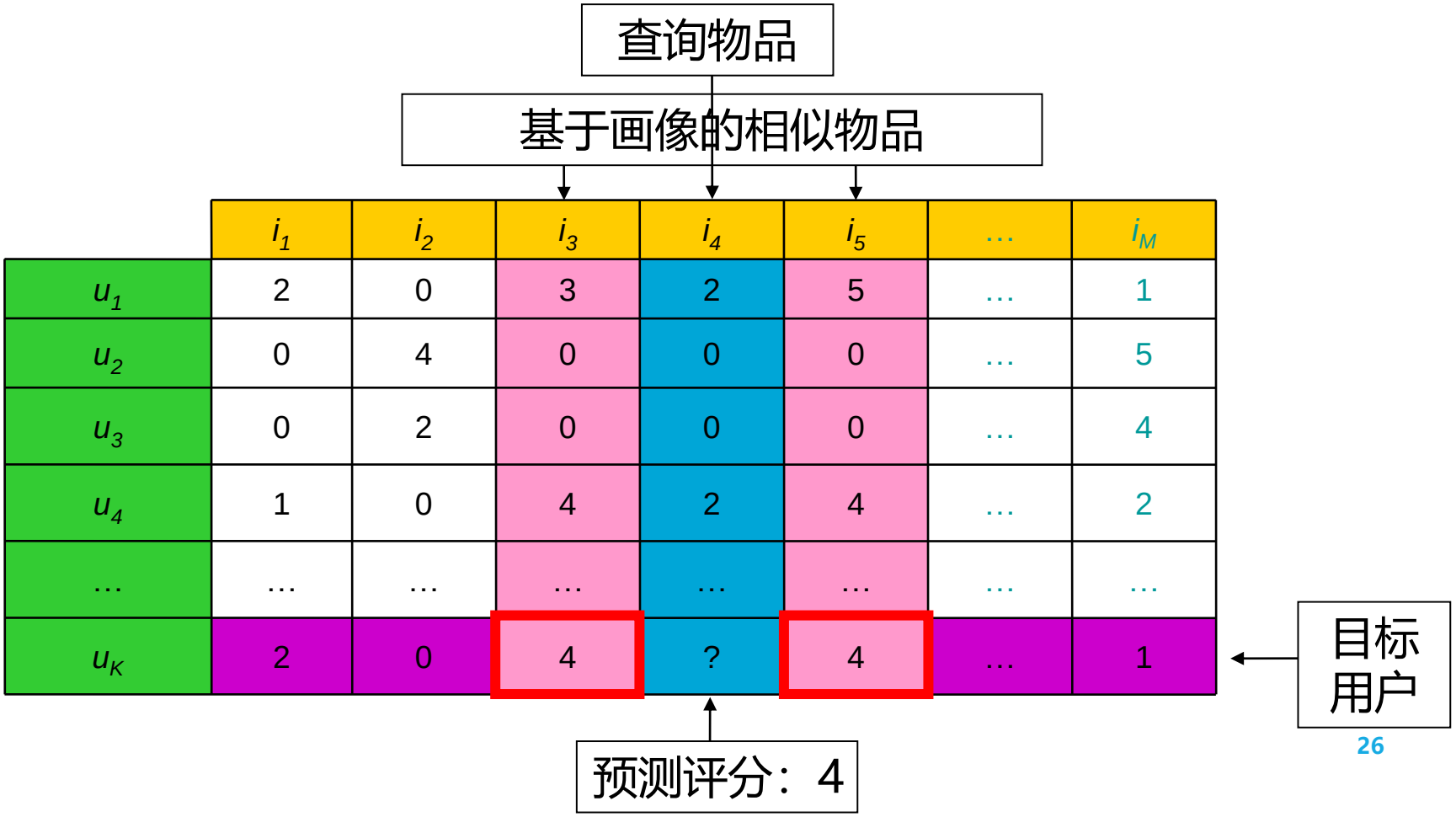
基于用户的KNN评分预测

- 预测从目标用户t与物品m之间的评分



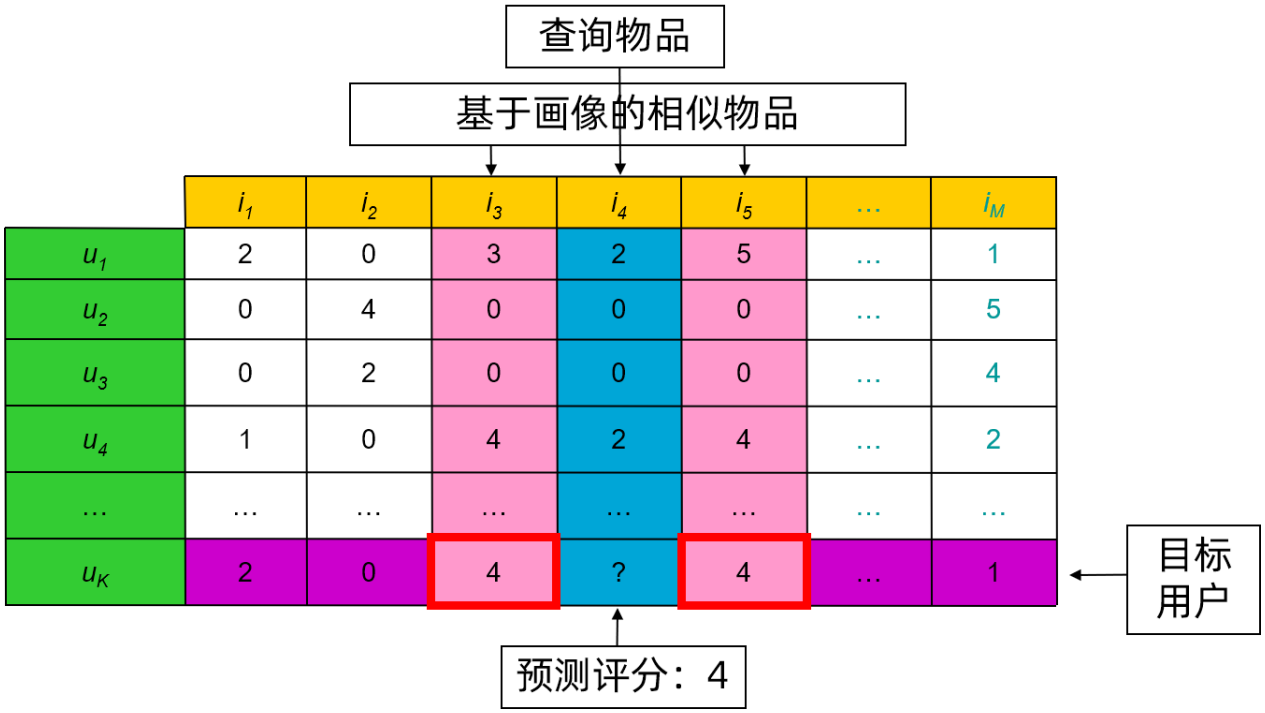
基于物品的KNN评分预测

- 基于物品相似度的推荐



基于物品的KNN评分预测

- 对于目标用户t的每个未评分物品m
  - 寻找类似物品{a}
  - 基于类似物品的集合{a}预测物品m的评分



相似度衡量（物品）

- 两个物品a和b之间的相似度量
  - 余弦（角）相似度

$$s_i^{cos}(i_a, i_b) = \frac{i_a^{\top} i_b}{\| i_a \| \| i_b \|} = \frac{\sum_u x_{u,a} x_{u,b}}{\sqrt{\sum_u x_{u,a}^2 \sum_u x_{u,b}^2}}$$

  - 调整后的余弦相似度

$$s_i^{adcos}(i_a, i_b) = \frac{\sum_m (x_{u,a} - \overline{x}_u)(x_{u,b} - \overline{x}_u)}{\sqrt{\sum_u (x_{u,a} - \overline{x}_u)^2 \sum_u (x_{u,b} - \overline{x}_u)^2}}$$

  - 皮尔森相关性

$$s_i^{corr}(i_a, i_b) = \frac{\sum_u (x_{u,a} - \overline{x}_a)(x_{u,b} - \overline{x}_b)}{\sqrt{\sum_u (x_{u,a} - \overline{x}_a)^2 \sum_u (x_{u,b} - \overline{x}_b)^2}}$$

基于物品的KNN评分预测

- 获取目标用户t评分过的k个最近邻物品

排序位置

$$N_i(u_t, i_a) = \{i_b | r_i(i_a, i_b) < K^*, x_{t,b} \neq 0\}$$

选择K^*，使$|N_i(u_t, i_a)| = k$

- 预测目标用户未评分的物品a的评分

$$\hat{x}_{t,a} = \frac{\sum_{i_b \in N_i(u_t, i_a)} s_i(i_a, i_b) x_{t,b}}{\sum_{i_b \in N_i(u_t, i_a)} s_i(i_a, i_b)}$$

由于使用目标用户本身数据进行预测，因此无需修正用户的平均评分

实证研究

- Movielens数据集来自 grouplens
  - https://grouplens.org/datasets/movielens/
  - 用户访问Movielens
    - 评分和收到电影的推荐
  - 数据集（ML-100k）
  - 100k个从1到5的评分
  - 943位用户，1682部电影（至少有一位用户评分）
  - 稀疏程度

$$1 - \frac{\#\text{non-zero entries}}{\text{total entries}} = 1 - \frac{10^5}{943 \times 1682} = 93.69\%$$

实验设置

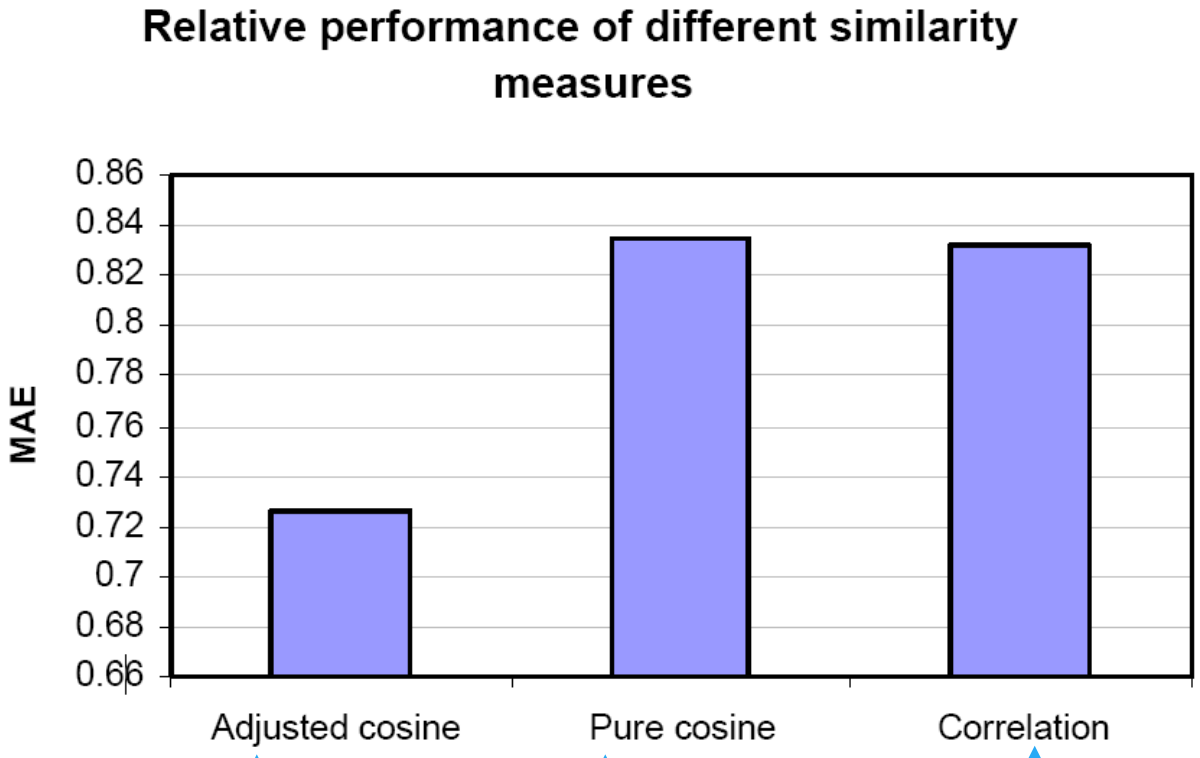
- 把数据拆分成训练集($x\%$)和测试集($(100-x)\%$)
  - 可重复T次，结果取平均值
- 评估指标
  - 平均绝对误差（MAE）

$$\text{MAE} = \frac{1}{|D_{\text{test}}|} \sum_{(u,i,r)\in D_{\text{test}}} |r - \hat{r}_{u,i}|$$

  - 均方根误差（RMSE）

$$\text{RMSE} = \sqrt{\frac{1}{|D_{\text{test}}|} \sum_{(u,i,r)\in D_{\text{test}}} \left(r - \hat{r}_{u,i}\right)^2}$$

相似度量的表现

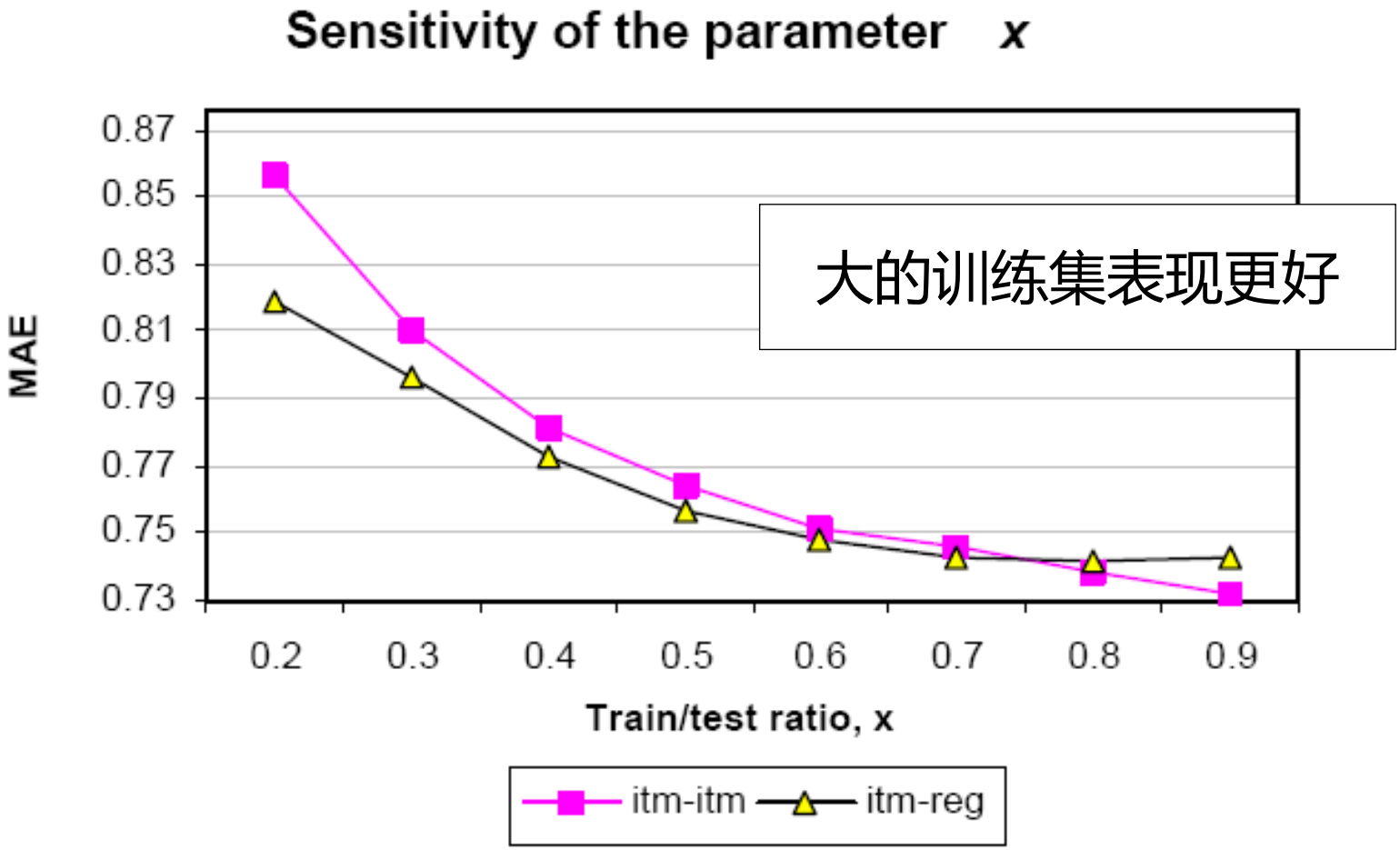


$$s_i^{adcos}(i_a, i_b) = \frac{\sum_m (x_{u,a} - \overline{x}_u)(x_{u,b} - \overline{x}_u)}{\sqrt{\sum_u (x_{u,a} - \overline{x}_u)^2 \sum_u (x_{u,b} - \overline{x}_u)^2}}$$

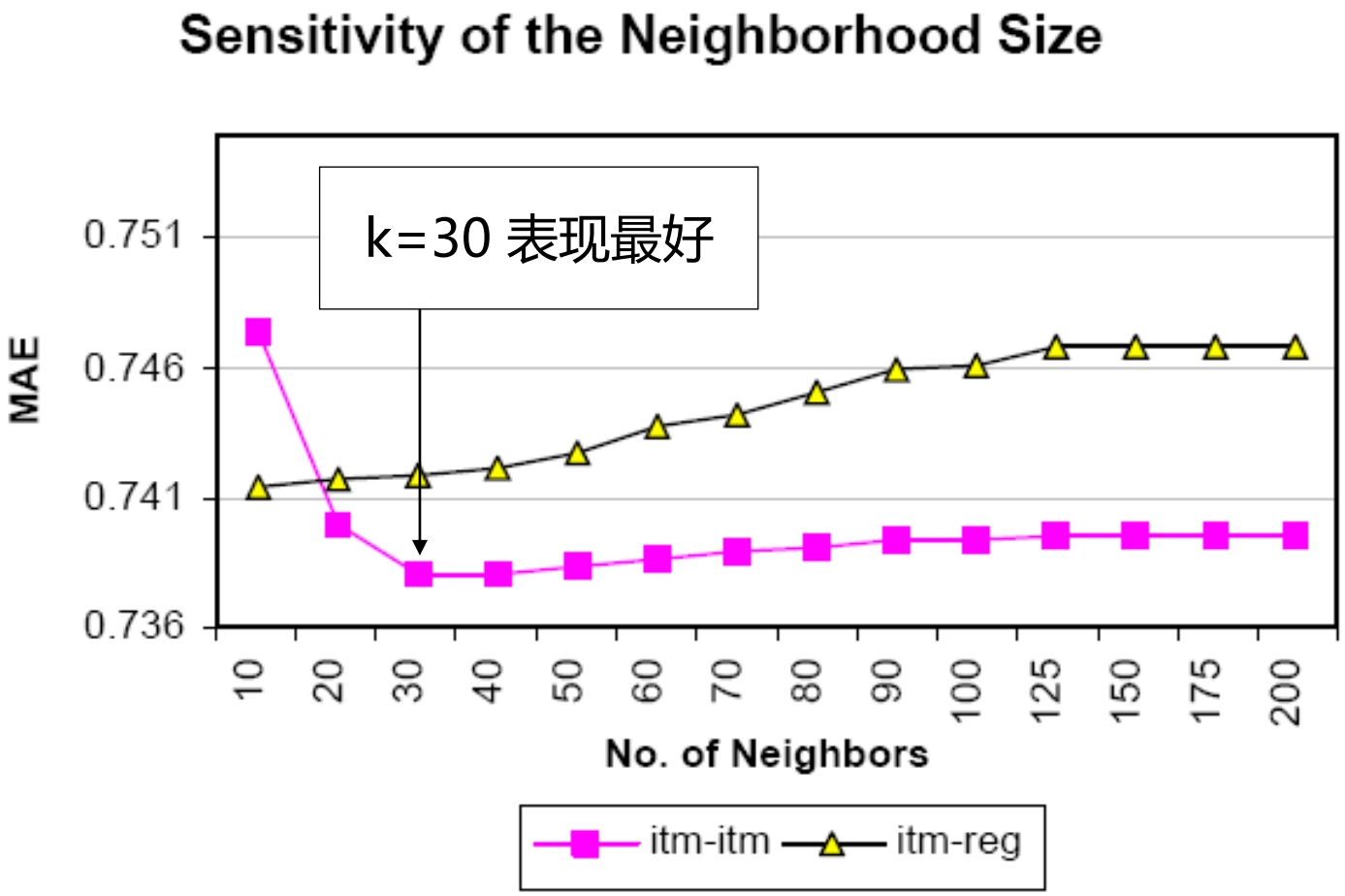
$$s_i^{corr}(i_a, i_b) = \frac{\sum_u (x_{u,a} - \overline{x}_a)(x_{u,b} - \overline{x}_b)}{\sqrt{\sum_u (x_{u,a} - \overline{x}_a)^2 \sum_u (x_{u,b} - \overline{x}_b)^2}}$$

$$s_i^{cos}(i_a, i_b) = \frac{\sum_u x_{u,a} x_{u,b}}{\sqrt{\sum_u x_{u,a}^2 \sum_u x_{u,b}^2}}$$

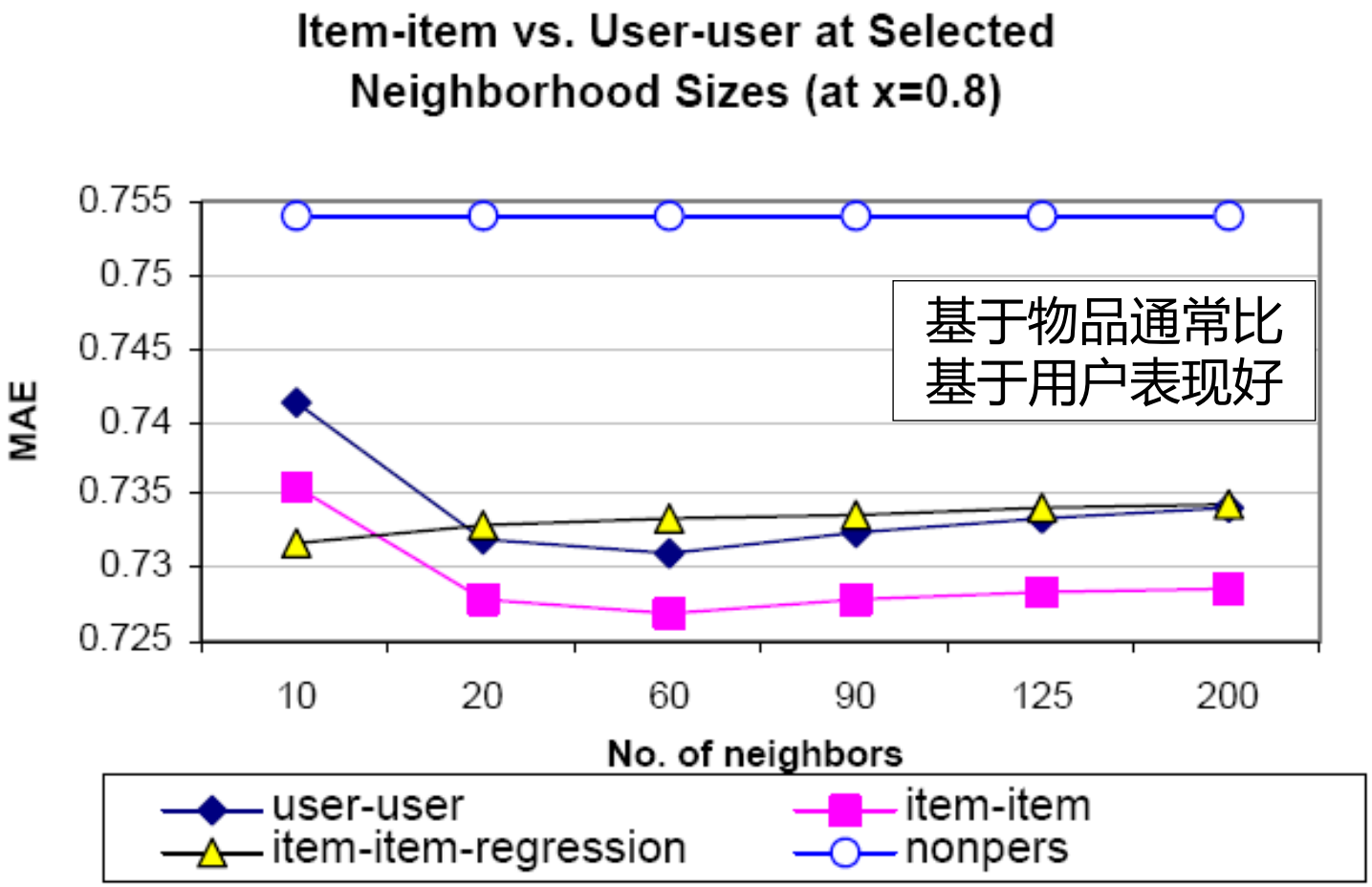
训练/测试比例的敏感度


Sensitivity of the parameter x
MAE
0.87
0.85
0.83
0.81
0.79
0.77
0.75
0.73
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
Train/test ratio, x
itm-itm
itm-reg
大的训练集表现更好

邻居大小k的敏感度


Sensitivity of the Neighborhood Size
k=30 表现最好
MAE
0.751
0.746
0.741
0.736
10
20
30
40
50
60
70
80
90
100
125
150
175
200
No. of Neighbors
itm-itm
itm-reg

基于物品 vs. 基于用户



- 物品-物品相似度通常更加稳定和客观

基于KNN的方法总结

- 直接，有高度可解释性
- 没有参数学习
  - 只有一个超参数k可以调整
  - 无法通过学习得到改善
- 效率可能是一个严重的问题
  - 当用户/项目编号很大时
  - 当有大量的用户-物品评分时
- 我们可能需要一个参数化和可学习的模型

k近邻算法

张伟楠 – 上海交通大学

k近邻算法 (K-Nearest Neighbor Algorithm)

- 用于分类和回归的非参数方法
  - 对于每个输入实例x，在特征空间中找到k个最接近的训练实例$N_k(x)$
  - 对于回归问题：对x的预测基于k个实例的标签的平均值

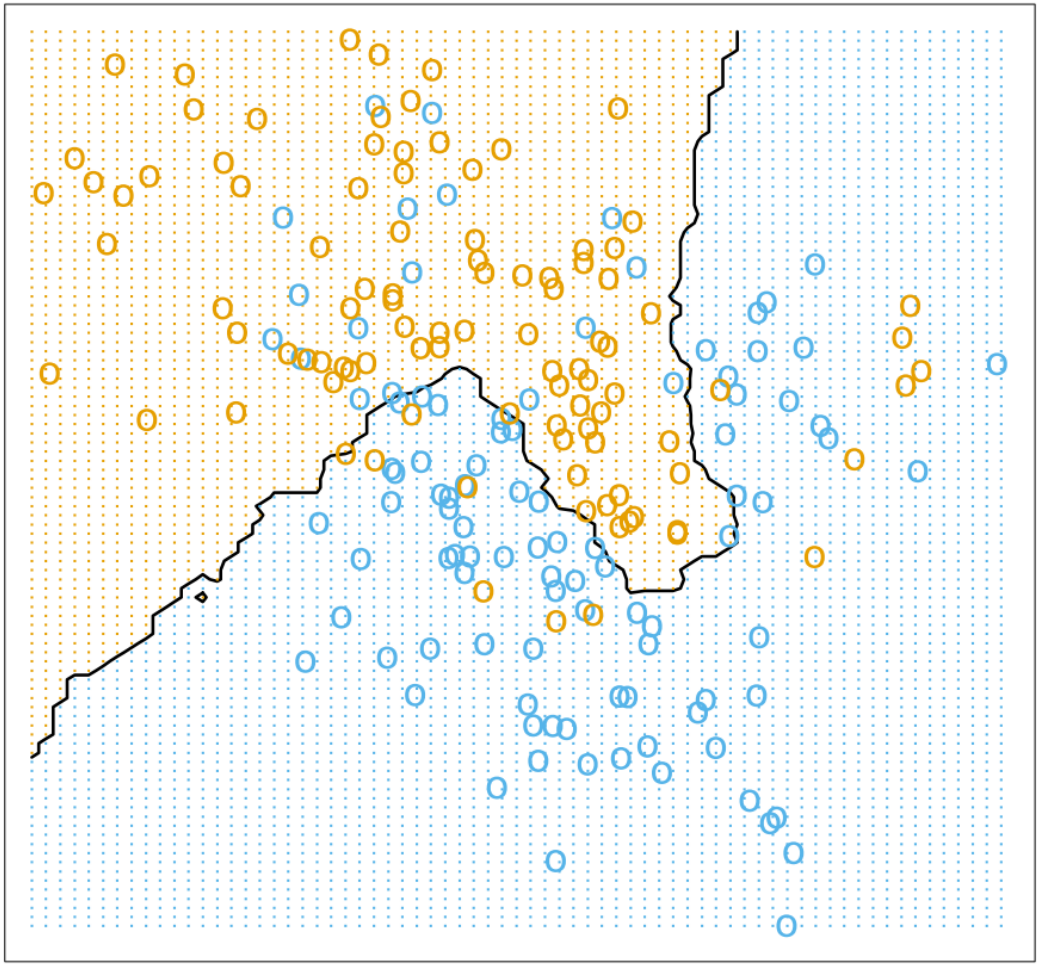
$$\hat{y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i$$

  - 对于分类问题：做邻居间的多数投票

$$\hat{y}(x) = \text{majority vote } \{y_i, x_i \in N_k(x)\}$$

KNN例子

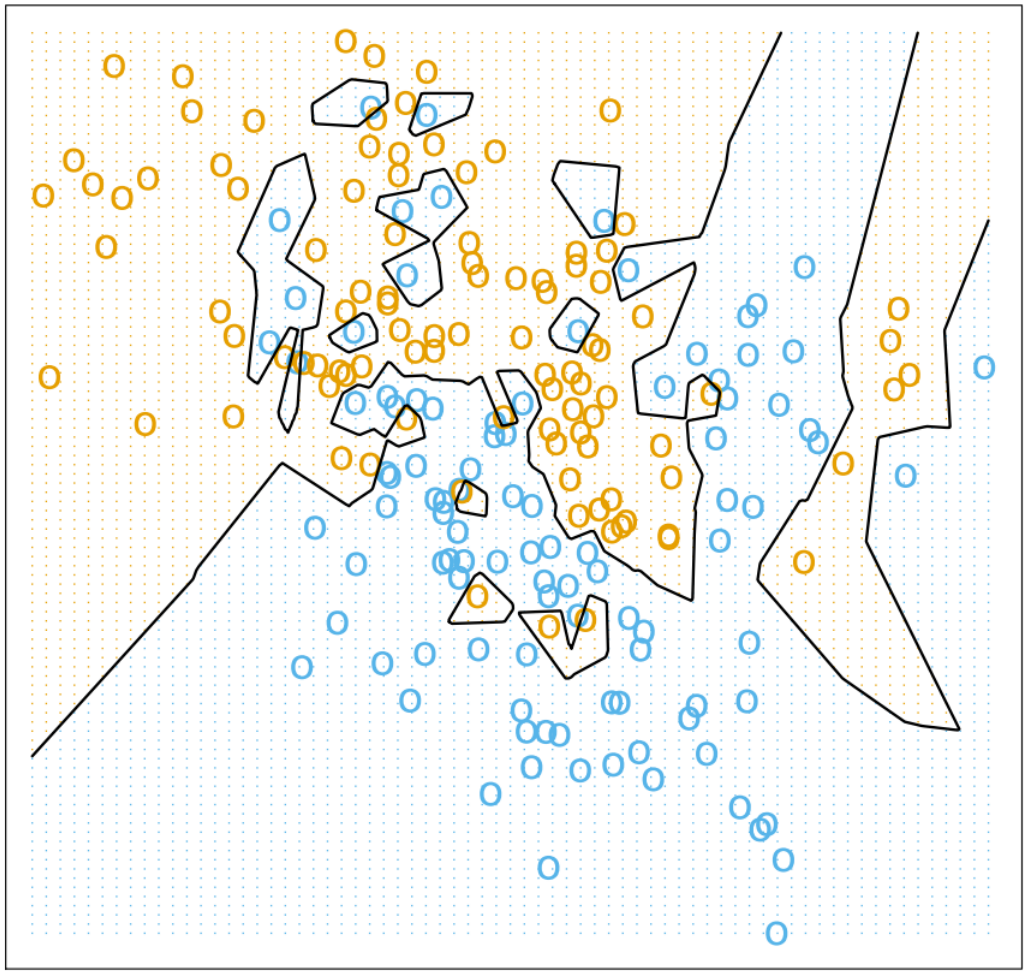
15-近邻



Jerome H. Friedman, Robert Tibshirani, and Trevor Hastie. “The Elements of Statistical Learning”. Springer 2009.

KNN例子

1-近邻



Jerome H. Friedman, Robert Tibshirani, and Trevor Hastie. “The Elements of Statistical Learning”. Springer 2009.

k近邻算法 (KNN)

- 广义版本
  - 定义输入实例x与其邻居x_i之间的相似度函数$s(x, x_i)$
  - 然后基于邻近相似性的加权标签平均值进行预测

$$\hat{y}(x) = \frac{\sum_{x_i \in N_k(x)} s(x, x_i) y_i}{\sum_{x_i \in N_k(x)} s(x, x_i)}$$

非参数KNN

- 没有参数需要学习
  - 实际上有N个参数：每个实例都是一个参数
  - 有N/k个有效参数
    - 直觉上讲，如果邻域不重叠，就会有N/k个邻域，每个邻域都对应一个参数
  - 超参数k
    - 我们不能将训练集上的平方误差和作为选择k的标准，因为$k = 1$时结果总是最好的
    - 在验证集上调整k的取值

k近邻思想在统计学习中的重要性

核方法

$$K\left(x^{(i)}, x^{(j)}\right) = \phi\left(x^{(i)}\right)^T \phi\left(x^{(j)}\right)$$

广义线性模型

$$\hat{\mathbf{y}} = \mathbf{\Phi}\hat{\boldsymbol{\theta}} = \mathbf{\Phi}\mathbf{\Phi}^T(\mathbf{\Phi}\mathbf{\Phi}^T + \lambda\mathbf{I}_n)^{-1}\mathbf{y}$$

$$= \mathbf{K}(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\mathbf{y}$$

$$\hat{\mathbf{y}} = \boldsymbol{\phi}(\boldsymbol{x})^T\hat{\boldsymbol{\theta}} = \mathbf{K}(\boldsymbol{x})(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\mathbf{y}$$

$$\mathbf{K}(\boldsymbol{x}) = \boldsymbol{\phi}(\boldsymbol{x})^T\mathbf{\Phi}^T$$

SVM模型

$$w^{*T}x + b^* = \left(\sum_{i=1}^{m} \alpha_i\, y^{(i)} \phi\left(x^{(i)}\right)\right)^T \phi(x) + b^*$$

$$= \sum_{i=1}^{m} \alpha_i\, y^{(i)} K\left(x^{(i)}, x\right) + b^*$$

- 统计学习其实就是基于两个数据点之间相似度$K\left(x^{(i)}, x^{(j)}\right)$
- 不同的模型基于这个相似度来做预测
- 神经网络模型可以学习这个相似度

矩阵分解

张伟楠 – 上海交通大学

矩阵分解（Matrix Factorization）

	Die Hard	Mission: Impossible	GoldenEye	Casino Royale	Titanic	Notting Hill	Bridget Jones's Diary	Love Actually
Boris	★★★☆☆	★★★★☆	★★★★★			★☆☆☆☆		★½☆☆☆
Dave		★★★★★	★★★★★	★★★★★				★☆☆☆☆
Will		★★☆☆☆			★★★★★	★★★★★	★★★☆☆	★★★★☆
George	★★★★☆	★★★★★	★★★★☆	★★★★☆				★★☆☆☆

≅

Boris	● ● ● ● ● ●
Dave	● ● ● ● ● ●
Will	● ● ● ● ● ●
George	● ● ● ● ● ●

·

Die Hard	Mission: Impossible	GoldenEye	Casino Royale	Titanic	Notting Hill	Bridget Jones's Diary	Love Actually
●	●	●	●	●	●	●	●
●	●	●	●	●	●	●	●
●	●	●	●	●	●	●	●
●	●	●	●	●	●	●	●
●	●	●	●	●	●	●	●
●	●	●	●	●	●	●	●

Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." Computer 42.8 (2009).

矩阵分解（Matrix Factorization）

物品

用户

1		3			5			5		4	
		5	4	?		4			2	1	3
2	4		1	2		3		4	3	5	
	2	4		5			4			2	
		4	3	4	2					2	5
1		3		3			2			4	

$$\hat{r}_{u,i} = p_u^{\top} q_i$$

i 物品

$\approx$ u 用户

.1	-.4	.2
-.5	.6	.5
-.2	.3	.5
1.1	2.1	.3
-.7	2.1	-2
-1	.7	.3

$\cdot$

1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4	-.9
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1	1.3
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6	.1

基本的矩阵分解模型

- 用户u在物品i上的预测

$$\hat{r}_{u,i} = p_u^\top q_i$$ ← 双线性模型

- 损失函数

$$\mathcal{L}(u, i, r_{u,i}) = \frac{1}{2}\left(r_{u,i} - p_u^\top q_i\right)^2$$

- 训练目标

$$\min_{P,Q} \sum_{r_{u,i} \in D} \frac{1}{2}(r_{u,i} - p_u^\top q_i)^2 + \frac{\lambda}{2}(\| p_u \|^2 + \| q_i \|^2)$$

- 梯度

$$\frac{\partial \mathcal{L}(u, i, r_{u,i})}{\partial p_u} = (p_u^\top q_i - r_{u,i}) q_i + \lambda p_u$$

$$\frac{\partial \mathcal{L}(u, i, r_{u,i})}{\partial q_i} = (p_u^\top q_i - r_{u,i}) p_u + \lambda q_i$$

带偏置的矩阵分解模型

- 用户u在物品i上的预测

$$\hat{r}_{u,i} = \mu + b_u + b_i + p_u^{\top} q_i$$

全局偏置　用户偏置　物品偏置　用户-物品交互

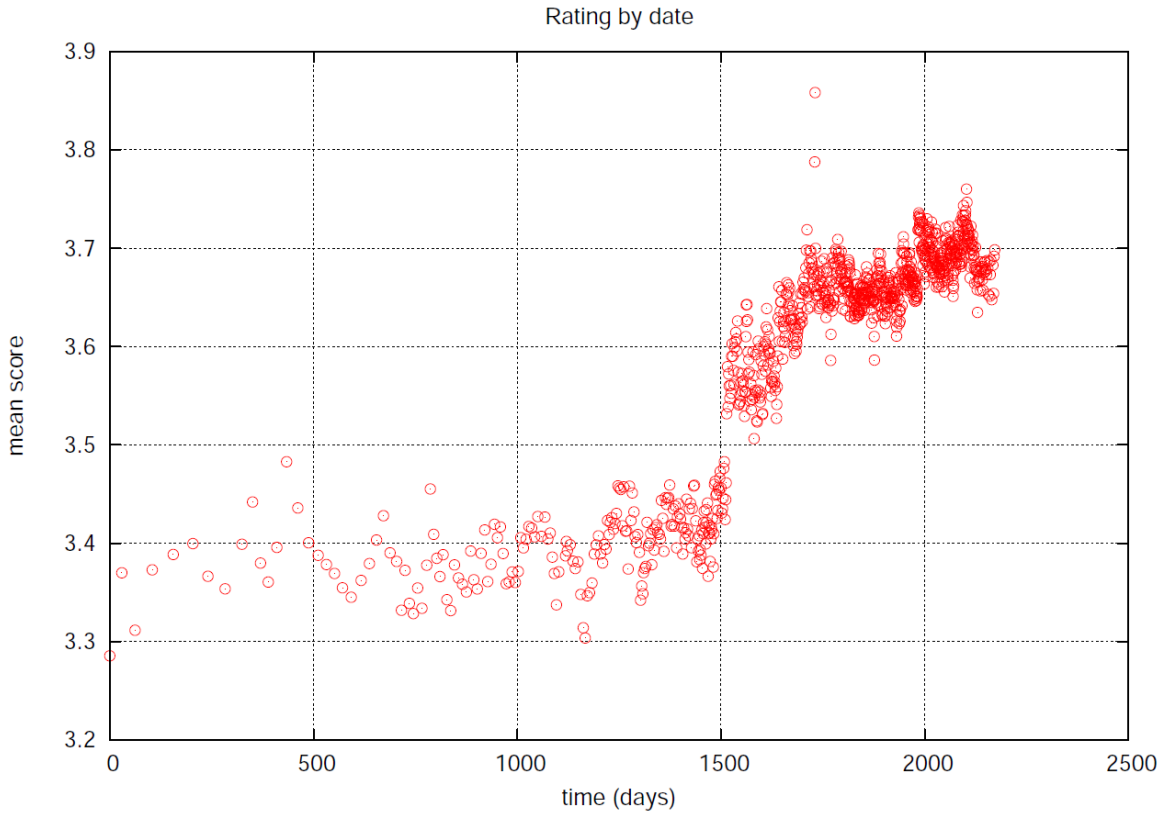
- 训练目标

$$\min_{P,Q} \sum_{r_{u,i} \in D} \frac{1}{2}(r_{u,i} - (\mu + b_u + b_i + p_u^{\top} q_i))^2 + \frac{\lambda}{2}(\| p_u \|^2 + \| q_i \|^2 + b_u^2 + b_i^2)$$

- 梯度更新

$$\delta = r_{u,i} - (\mu + b_u + b_i + p_u^{\top} q_i)$$
$$\mu \leftarrow \mu + \eta\delta$$
$$b_u \leftarrow (1 - \eta\lambda) b_u + \eta\delta$$
$$b_i \leftarrow (1 - \eta\lambda) b_i + \eta\delta$$
$$p_u \leftarrow (1 - \eta\lambda) p_u + \eta\delta q_i$$
$$q_i \leftarrow (1 - \eta\lambda) q_i + \eta\delta p_u$$

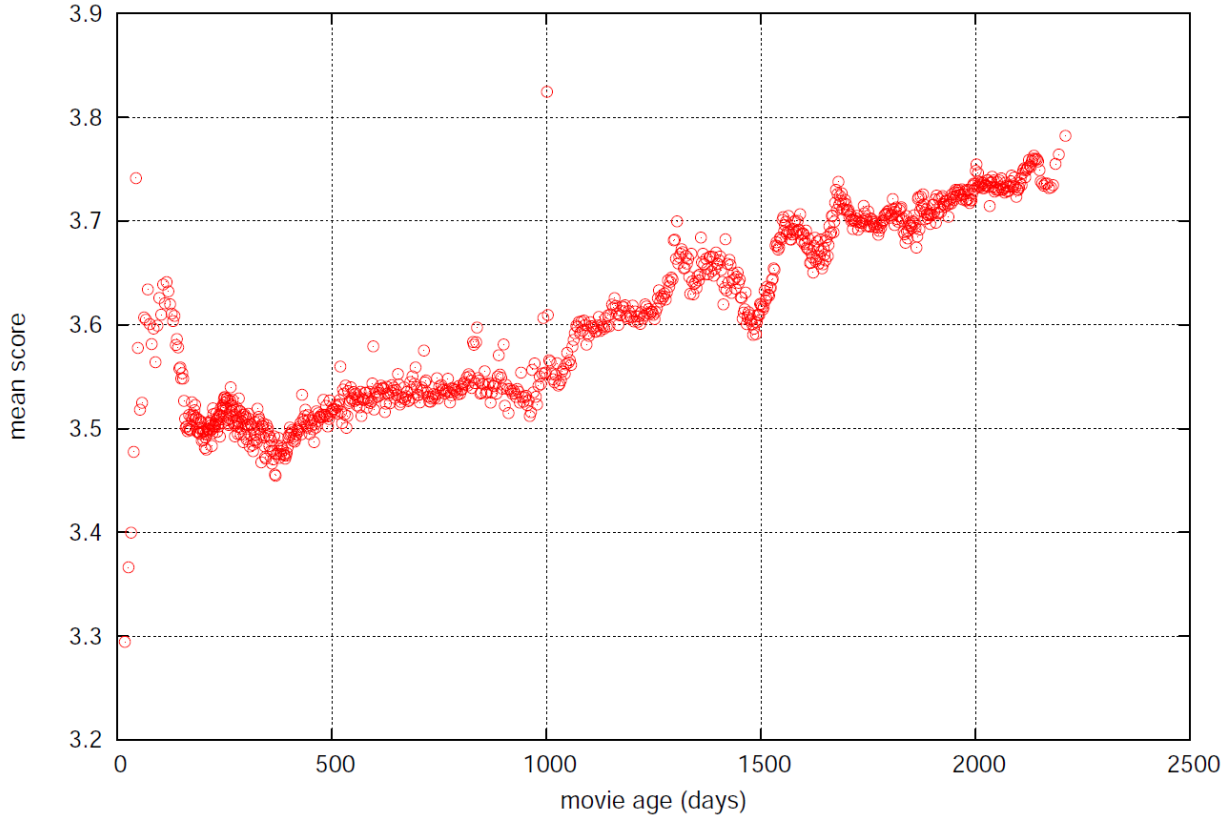
时间动态变化



- 约1500天（2004年初）进入数据集的平均电影评分突然上升

Koren, Yehuda. "Collaborative filtering with temporal dynamics. " KDD 2009.

时间动态变化



- 对于更老的电影，人们往往给予更高的评分

Koren, Yehuda. "Collaborative filtering with temporal dynamics. " KDD 2009.

动态变化的多个原因

- 物品方面的影响
  - 物品受欢迎程度不断变化
  - 季节性原因影响物品的受欢迎程度
- 用户方面的影响
  - 用户改变了他们的品味
  - 暂时的，短期偏置
  - 浮动的评分等级
  - 在一个家庭里实际评分人的改变

解决动态变化问题

- 因子模型方便地允许分别处理不同原因
- 我们观察到的变化：
  - 个人用户的评分等级$b_u(t)$
  - 个人物品的受欢迎程度$b_i(t)$
  - 用户偏好$p_u(t)$

$$r_{u,i}(t) = \mu + b_u(t) + b_i(t) + p_u(t)^{\mathsf{T}} q_i$$

- 设计指南
  - 物品显示较慢的时间变化
  - 用户表现出频繁和突然的变化
  - 因子$p_u(t)$的建模成本很高
  - 通过大量参数化函数来获得灵活性

Review: 带偏置的矩阵分解模型

- 用户u在物品i上的预测

$$\hat{r}_{u,i} = \mu + b_u + b_i + p_u^\top q_i$$

全局偏置　用户偏置　物品偏置　用户-物品交互

- 训练目标

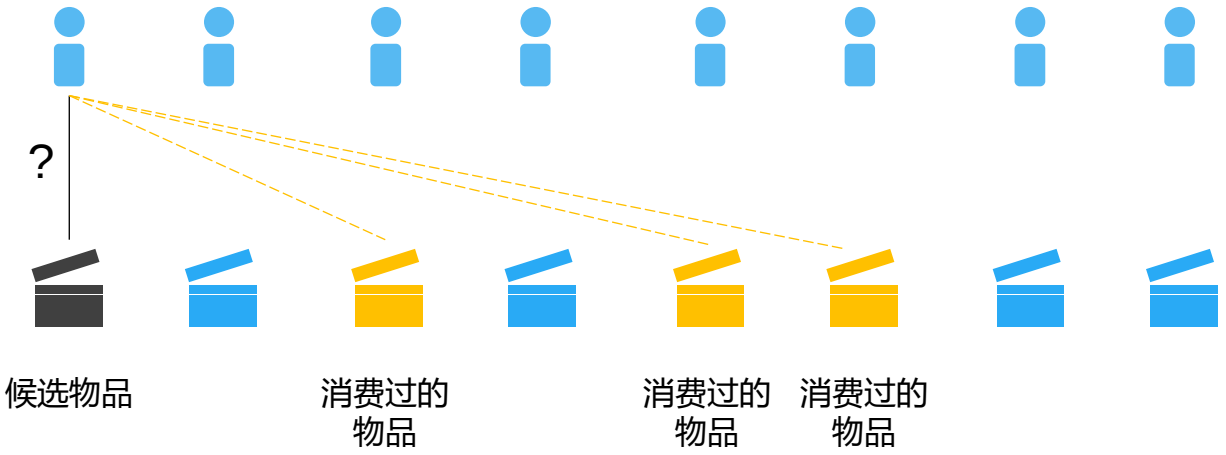
$$\min_{P,Q} \sum_{r_{u,i} \in D} \frac{1}{2}(r_{u,i} - (\mu + b_u + b_i + p_u^\top q_i))^2 + \frac{\lambda}{2}(\| p_u \|^2 + \| q_i \|^2 + b_u^2 + b_i^2)$$

- 梯度更新

$$\begin{aligned}
\delta &= r_{u,i} - (\mu + b_u + b_i + p_u^\top q_i) \\
\mu &\leftarrow \mu + \eta\delta \\
b_u &\leftarrow (1 - \eta\lambda) b_u + \eta\delta \\
b_i &\leftarrow (1 - \eta\lambda) b_i + \eta\delta \\
p_u &\leftarrow (1 - \eta\lambda) p_u + \eta\delta q_i \\
q_i &\leftarrow (1 - \eta\lambda) q_i + \eta\delta p_u
\end{aligned}$$

基于邻居（相似性）的矩阵分解

- 假设：用户以前消费过的物品反映了她的品味
- 从这些“类似的”物品中获取未知评分（物品-物品变体）



基于邻居的矩阵分解：SVD++

$$\hat{r}_{u,i} = b_{u,i} + q_i^{\top}(p_u + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} y_j)$$

- 需要估计用户u的评分物品i
- 偏置
- 物品向量
- 用户向量
- 用户之前评分的物品的集合
- 为了表示邻居相似度的物品向量

- 每个物品有两个潜在向量
  - 本身的物品向量q_i
  - 用于估计候选物品和目标用户之间的相似度的向量y_i

Koren, Yehuda. "Factorization meets the neighborhood: a multifaceted collaborative filtering model." KDD, 2008.

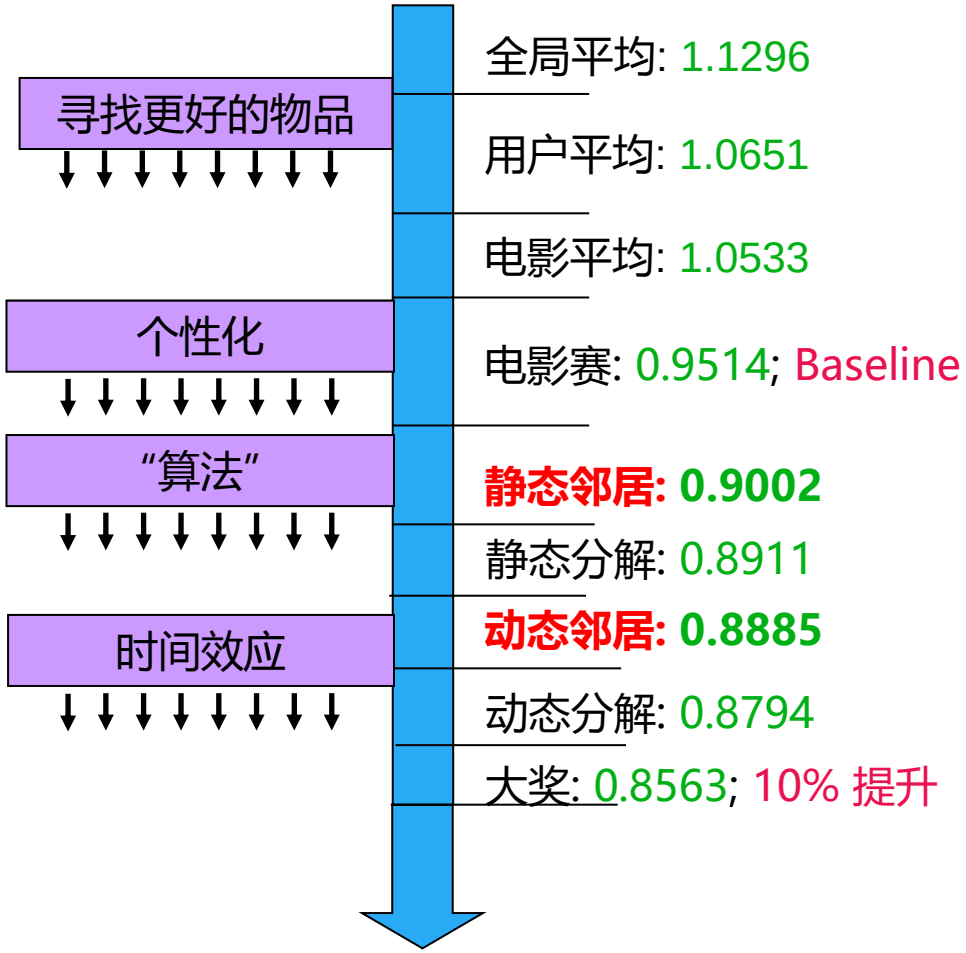
Netflix奖

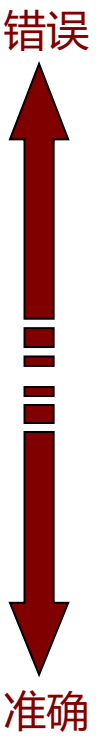
- 针对电影的最佳协同过滤算法的公开比赛
  - 始于2006年10月2日
  - 一项价值数百万美元的挑战，将Netflix推荐算法的准确度（RMSE）提高10%

- Netflix提供
  - 训练数据：100,480,507评分：
  - 480,189位用户x 17,770部电影
  - 格式：<用户，电影，日期，评级>

- 两种流行的方法：
  - 矩阵分解
  - K近邻算法

Netflix奖





动态邻居模型作为动态因子模型提供相同的相对RMSE提升(0.0117)


NETFLIX
2009
DATE 09.21.09
PAY TO THE ORDER OF BellKor's Pragmatic Chaos
$ 1,000,000 00
AMOUNT ONE MILLION 00/100
FOR The Netflix Prize
Reed Hastings
The Netflix Prize

双线性模型

张伟楠 – 上海交通大学

双线性

- 在数学中，双线性的含义为：二元函数固定任意一个自变量时，函数关于另一个自变量线性。
- 具体地，二元函数 $f: \mathbb{R}^n \times \mathbb{R}^m \mapsto \mathbb{R}^l$ 是双线性函数，当且仅当对任意 $u, v \in \mathbb{R}^n$ 和 $s, t \in \mathbb{R}^m$ 满足：

1. $f(u, s+t) = f(u, s) + f(u, t)$
2. $f(u+v, s) = f(u, s) + f(v, s)$
3. $f(u, \lambda s) = \lambda f(u, s)$
4. $f(\lambda u, s) = \lambda f(u, s)$

- 最简单的双线性函数的例子是向量内积 $\langle \cdot, \cdot \rangle$，可以验证：

1. $\langle u, s+t \rangle = \langle u, s \rangle + \langle u, t \rangle$
2. $\langle u+v, s \rangle = \langle u, s \rangle + \langle v, s \rangle$
3. $\langle u, \lambda s \rangle = \lambda \langle u, s \rangle$
4. $\langle \lambda u, s \rangle = \lambda \langle u, s \rangle$

T. Chen et al. Feature-based matrix factorization. arXiv:1109.2271
Open source: https://apex.sjtu.edu.cn/projects/33

矩阵分解是最简单的双线性模型之一

物品

用户

1		3			5			5		4	
		5	4	?		4			2	1	3
2	4		1	2		3		4	3	5	
	2	4		5			4			2	
		4	3	4	2					2	5
1		3		3			2			4	

$$\hat{r}_{u,i} = p_u^\top q_i$$

$\approx$

用户 u

.1	-.4	.2
-.5	.6	.5
-.2	.3	.5
1.1	2.1	.3
-.7	2.1	-2
-1	.7	.3

$\cdot$

i 物品

1.1	-.2	.3	.5	-2	-.5	.8	-.4	.3	1.4	2.4	-.9
-.8	.7	.5	1.4	.3	-1	1.4	2.9	-.7	1.2	-.1	1.3
2.1	-.4	.6	1.7	2.4	.9	-.3	.4	.8	.7	-.6	.1

因子分解机

张伟楠 – 上海交通大学

基于特征的矩阵分解

$$\hat{y} = \mu + \left(\sum_j b_j^{(g)}\gamma_j + \sum_j b_j^{(u)}\alpha_j + \sum_j b_j^{(i)}\beta_j\right) + \left(\sum_j p_j\alpha_j\right)^{\mathsf{T}}\left(\sum_j q_j\beta_j\right)$$

- 将所有信息视为特征
  - 用户id和商品id
  - 时间、商品类别、用户统计资料等
- 用户和商品特征具有潜在因子

T. Chen et al. Feature-based matrix factorization. arXiv:1109.2271
Open source: https://apex.sjtu.edu.cn/projects/33

因子分解机(Factorization Machine)

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d-1} \sum_{j=i+1}^{d} \langle v_i, v_j \rangle x_i x_j$$

- 每个离散（分类）字段使用独热编码
- 每个连续字段使用实值特征
- 所有特征都有潜在因子
- 更普适的回归模型

Steffen Rendle. Factorization Machines. ICDM 2010
https://www.ismll.uni-hildesheim.de/pub/pdfs/Rendle2010FM.pdf
Open source: http://www.libfm.org/

因子分解机 (Factorization Machine)

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d-1} \sum_{j=i+1}^{d} \langle v_i, v_j \rangle x_i x_j$$

- 再对参数v_s求梯度的结果为

为了简洁，我们采用了不太严谨的写法，当s=1或s=d时会出现求和下界大于上界的情况

$$\begin{aligned}
\nabla_{\boldsymbol{v}_s} \hat{y} &= \nabla_{\boldsymbol{v}_s} \left(\sum_{i=1}^{d-1} \sum_{j=i+1}^{d} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle x_i x_j \right) \\
&= \nabla_{\boldsymbol{v}_s} \left(\sum_{j=s+1}^{d} \langle \boldsymbol{v}_s, \boldsymbol{v}_j \rangle x_s x_j + \sum_{i=1}^{s-1} \langle \boldsymbol{v}_i, \boldsymbol{v}_s \rangle x_i x_s \right) \\
&= x_s \sum_{j=s+1}^{d} x_j \boldsymbol{v}_j + x_s \sum_{i=1}^{s-1} x_i \boldsymbol{v}_i \\
&= x_s \sum_{i=1}^{d} x_i \boldsymbol{v}_i - x_s^2 \boldsymbol{v}_s
\end{aligned}$$

Steffen Rendle. Factorization Machines. ICDM 2010
https://www.ismll.uni-hildesheim.de/pub/pdfs/Rendle2010FM.pdf
Open source: http://www.libfm.org/

因子分解机 (Factorization Machine)

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d-1} \sum_{j=i+1}^{d} \langle v_i, v_j \rangle x_i x_j$$

- 上式时间复杂度为$O(kd^2)$，可以做如下变形，将复杂度降至$O(kd)$

$$\begin{aligned}
\sum_{i=1}^{d-1} \sum_{j=i+1}^{d} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle x_i x_j &= \frac{1}{2} \left(\sum_{i=1}^{d} \sum_{j=1}^{d} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle x_i x_j - \sum_{i=1}^{d} \langle \boldsymbol{v}_i, \boldsymbol{v}_i \rangle x_i^2 \right) \\
&= \frac{1}{2} \left(\sum_{i=1}^{d} \sum_{j=1}^{d} \langle x_i \boldsymbol{v}_i, x_j \boldsymbol{v}_j \rangle - \sum_{i=1}^{d} \langle x_i \boldsymbol{v}_i, x_i \boldsymbol{v}_i \rangle \right) \\
&= \frac{1}{2} \left\langle \sum_{i=1}^{d} x_i \boldsymbol{v}_i, \sum_{j=1}^{d} x_j \boldsymbol{v}_j \right\rangle - \frac{1}{2} \sum_{i=1}^{d} \langle x_i \boldsymbol{v}_i, x_i \boldsymbol{v}_i \rangle \\
&= \frac{1}{2} \sum_{l=1}^{k} \left(\sum_{i=1}^{d} v_{il} x_i \right)^2 - \frac{1}{2} \sum_{l=1}^{k} \sum_{i=1}^{d} v_{il}^2 x_i^2
\end{aligned}$$

Steffen Rendle. Factorization Machines. ICDM 2010
https://www.ismll.uni-hildesheim.de/pub/pdfs/Rendle2010FM.pdf
Open source: http://www.libfm.org/

实验结果（review线性模型最后部分的实验）

- 性能

Model	Linearity	AUC		Log Loss	
		Criteo	iPinYou	Criteo	iPinYou
Logistic Regression	Linear	71.48%	73.43%	0.1334	5.581e-3
Factorization Machine	Bi-linear	72.20%	75.52%	0.1324	5.504e-3
Deep Neural Networks	Non-linear	75.66%	76.19%	0.1283	5.443e-3

- 与非线性模型相比，线性模型具有以下优缺点
  - 优点：标准化，易于理解和实施，高效和可扩展
  - 缺点：建模局限（特征独立假设），无法探索特征交互

总结KNN和双线性模型

- 协同过滤是推荐系统中的重要机器学习技术，主要探索学习数据点之间的相似性，进而对目标标签做出预测
- KNN算法是最简单的非参数化模型
- 矩阵分解是最基础的双线性模型。参数向量的内积操作是双线性模型中最常使用的操作，这时对一边参数向量求梯度，得到另一边参数向量的取值
- 因子分解机是一种通过向量内积来直接学习两两特征交互对标签预测的影响的通用模型

THANK YOU